Towards a Generic Multidisciplinary Models Composition Tool

Anas Abouzahra, Ayoub Sabraoui and Karim Afdel
Laboratory of Computer Systems and Vision LabSIV, Faculty of Science, Ibn Zohr University, Agadir, Morocco

1 RESEARCH PROBLEM

Emergent engineering domains like healthcare engineering, neural engineering, financial engineering and many other domains get developed more and more quickly, pushing the systems complexity to an inexorable growth. In this particular context, modeling and especially MDE (Model Driven Engineering) approach (Mellor, 2003), plays an important role today. In fact, modeling reduces the increasing complexity of systems, mainly by reducing the gap between the problems treatment and the technologies spaces (France, 2007) (Shmidt, 2006). It provides domain-specific software tools to non-programmer engineers to take advantage of using computer computing power, by building themselves specific softwares for their domains.

A project's complexity can be defined in terms of variability and interdependency (Baccarini, 1987). These two dimensions are particularly important and cover a number of current industrial challenges. When constructing a rocket for example, one has to switch between thermal, mechanical, electrical, and many more aspects. Each of these aspects is complex by itself. But even more complex are the numerous relations between these various sub-systems. It seems obvious that the need to express relations between multidisciplinary models is a reality that we can no longer escape.

In this paper we present a generic multidisciplinary models composition tool project; a new contribution to address multidisciplinary MDE development needs. We start in Section 2 by giving a quick overview of the outline of the objectives. Then we present a concise state of the art listing the related works in Section 3. Section 4 describes our methodology. After this, in Section 5, we present the expected outcomes. Section 6 exposes the actual stage of the research. The concluding section summarizes the usefulness and provides future perspectives of our work.

2 OUTLINE OF OBJECTIVES

When you want to develop an application with a MDE approach, it became practically a necessity to use tools provided by the Eclipse Modeling Framework (EMF) project (EMF, 2015) or tools around these technologies, especially, if you want to respect the standards defined by the Object Management Group (OMG). The overall goal of this project is to propose a new solution to contribute to open modeling tools based on the EMF to multidisciplinary development context. Our contribution will consist on three complementary modules. The first one is called KM4M (Kernel Metametamodel for Modellaborate). It is a small and concise textual language for describing metamodels accompanied by a rich text editor. It will provide requisite expressions to define relations between heterogeneous metamodels. These relations express the potential interlinks between models conform to these metamodels. The second module is called CG4M (Code Generation for Modellaborate). It is a code generation facility which is capable of generating every needed Java interfaces and implementations for all the classes in a metamodel described by KM4M language, including methods to instantiate and manage model interlinks. This generation will be done on the fly or on demand in connection with the metamodel editor. The last module is called MR4M (Metamodels Repository for Modellaborate). It is a metamodel repository that communicates with the other two modules to facilitate editing of metamodels and provides the necessary information for the automatic code generation. The three modules will work together in order to be able to set up links between EMF heterogeneous models (by heterogeneous we mean conforming to distinct metamodels). All of that with a synchronisation mechanism that diffuses models modifications each time that a model has been changed.

Abouzahra, A., Sabraoui, A. and Afdel, K.
Towards a Generic Multidisciplinary Models Composition Tool.
In *Doctoral Consortium (DCMODELSWARD 2016)*, pages 9-15

3 STATE OF THE ART

The impact of MDE can be measured solely through its actual applications in concrete use cases in software industry. These measures are highly correlated with the power and the limits of technology available. Actually, the OMG has defined some standards around the MDE: among other, the Meta Object Facility (MOF 2.5). The MOF 2.5 (MOF 2.5, 2015) is at the top layer of modeling pyramid as a meta-metamodeling language for defining metamodels. The MOF 2.5 plays exactly the role that EBNF (a notation to define grammars) (Pattis, 1980) plays for defining programming language grammars. The OMG has defined also the EMOF 2.5 (Essential MOF 2.5) (MOF 2.5, 2015), a simplified version of the MOF 2.5. In parallel, The OMG has defined the XML Metadata Interchange (XMI 2.5); a standard for exchanging metadata (XMI 2.5, 2015). It is thereafter the by default persistence format of models conforming to MOF 2.5. In this wake, the EMF project has provided Ecore (EMF, 2015); the core metamodel at the heart of EMF that implements the EMOF 2.5. The Ecore is today the main reference implementation of the EMOF 2.5. The EMF project provides in addition a set of tools and runtime support to produce code generation from a model specification described in XMI (Steinberg, 2009) (EMF, 2015). The EMF project became the de facto set of tools to build applications according to the MDE approach. However, EMF has shown clear limits in terms of flexibility (Bagnato, 2014) (Fouquet, 2012), scalability (Kolovos 2013) (Barmpis, 2012) (Benelallam, 2014) and extensibility (Garmendia, 2014), especially when was actually put to the test on complex systems context.

In order to overcome these limitations, a substantial number of research works were carried out, focusing essentially on building layers on top of EMF. Model composition has been the subject of great deal of projects. We could mention some of the important ones: Epsilon Merging Language; EML (Kolovos, 2009), Kompose (Fleurey, 2007), Atlas Model Weaver; AMW (Didonet, 2009) and Virtual EMF; VEMF (Clasen, 2011). The EML language is a hybrid rule based language for merging models. This approach proposes to merge homogeneous or heterogeneous models through three categories of rules (Match, Merge, and Transform). It reuses the syntax and semantics of ETL (Epsilon transforming language) (Kolovos, 2008) and extends it with concepts specific to model merging. Kompose implements a generic structural composition operator that can be specialized to a particular modeling language described by a metamodel. The composition mechanism is done in two steps: matching the model elements that describe the same concepts, then merging those elements. The Atlas Model Weaver is a model composition framework that uses model weaving and model transformation to define and execute composition operation. The model weaving captures the links between model elements. Then, a transformation produces the composed model. Virtual EMF is a model composition approach based on the AMW project and providing a virtualization mechanism that offers a direct and transparent access to the contributing models used in the composition process. The weaving model becomes a virtual model.

To enable a synthesis to assess the relevance of each approach, we used the set of following criteria:

- **Heterogeneity**: it means that the solution is able to provide composition of heterogeneous models.
- **Number of Input Models:** since we are interested to multidisciplinary modeling context, this criterion characterizes the possible limitation of the number of input models.
- **Symmetric/ Asymmetric:** These two properties distinguish symmetric approaches; where no distinction is made between source models, from asymmetric approaches; where base models play the major role in the composition, when aspect models provide weaving concepts.
- **Synchronization:** This criterion refers to the ability of the approach to spread the changes made in the source models into the composed ones.

Table 1: Evaluation of composition approaches.

Approach / Criteria	EML	Kompose	AMW	VEMF
Heterogen	Yes	No	Yes	Yes
Nb of Inputs	No limit	2	No limit	No limit
Sym / Asym	S	A	S	S
Synch	No	No	No	Yes

Table “Tab. 1” provides an overview of evaluating the approaches against the defined criteria. Except Virtual EMF, the other approaches rely on the use of a third model combined with some data redundancy. In plus, they do not provide automatic synchronization with source models.

Virtual EMF has presented a promising approach but it remained at its prototyping stage. We will not criticize here these approaches, since they have demonstrated their usefulness in many use cases. What we can note is that evaluation shows that they are less adapted to a multidisciplinary development context, where heterogeneous models must collaborate.

4 METHODOLOGY

To illustrate the approach proposed in this paper, we use an example in which two disciplines interact together. It is about the calculation of employee benefits according to the International Accounting Standard Nineteen IAS 19 (IASB 19, 2004). This example will be explained below in Section 6.

To prove our work we decide to follow a methodology in five steps:

- Develop a first prototype of our case study using conventional MDE tools (EMF models, XMI serialization ... etc.).
- Analyse the advantages of using modeling techniques to the development of a Domain Specific Modeling Tool (DSMT) before showing the limitations induced inter alia by the lack of a powerful heterogeneous models composition mechanism.
- Develop a prototype of our solution for multidisciplinary models composition.
- Rebuild the first DSMT prototype with our proposal.
- Finally, demonstrate the improvements that our solution brings.

5 EXPECTED OUTCOME

The solution we intend to implement is expected to provide a model composition mechanism capable to link elements from heterogeneous models. This will occur through three modules: KM4M, CG4M and MR4M.

5.1 KM4M: A Rich Language for Metamodels Specification

The MDE challenges to drive development of rich domain specific tools that evolve into a complex multidisciplinary context. Our contribution has been designed to complete existing tools to provide a multidisciplinary models composition mechanism. Since EMOF 2.5 does not include concepts to express relations between different metamodels, we propose to overcome this limitation by extending the EMOF 2.5 concepts by introducing new relation concepts that can be expressed between metamodels. Then, we propose a new language named KM4M that allows specifying domain definition metamodels and provides some relatively simple features to describe their relationships. KM4M is therefore a Domain Specific Language (DSL) to define metamodels. The KM4M itself has its own metamodel. It is a meta-metamodel, to which other domain metamodels conform. This meta-metamodel may be defined in KM4M, just like EBNF may be defined in EBNF using only few lines. It uses concepts like Class, Attribute, and Reference. It is structurally close to EMOF 2.5 with additional concepts to express relations between metamodels. We plan to implement KM4M as an Eclipse plug-in. It will be accompanied by a rich editor and will work in collaboration with two other modules that we present in the paragraphs to follow.

5.2 CG4M: An Automatic Code Generation Module

The CG4M module will be a code generation facility able to automatically produce all necessary code to instantiate and enrich models. It will generate Java interfaces and implementation classes for all the classes defined by a metamodel, a factory and package implementation class, and adapter classes that adapt metamodel classes to manage inter-model composition links. The goal is to be able to instantiate composition links conforming to relationships defined in the metamodels and use them in a fully transparent way as if we were using an intra-model link (links between elements of the same model).

All entities of EMF models are subclasses of the super class *EObject* which provide a set of methods to access an entity. Among these are reflective methods *eSet()* and *eGet()* to access its attributes and references. CG4M will provide a specific implementation of *EObject* class that allows using those methods to navigate transparently a composition link.

We plan to implement CG4M as an Eclipse plug-in. It will produce code generation automatically from metamodels written with KM4M, on the fly by default, or by manual activation if the user chooses a separate generation mode.

5.3 MR4M: A Metamodels Repository

The MR4M module will be an Eclipse plug-in for providing metamodels to other tools and services dealing with models.

For each platform, we suppose that there is an associated metadata repository defining the metadata associated to this platform. Within the content of such a platform, the metadata repository records all available resources. In the context of a MDE multidisciplinary development platform, a repository of metamodels from several disciplines will make available, in such a platform (in particular, if it integrates the two previous modules: KM4M and CG4M), all the necessary information to build their metamodels and use composition links between models.

6 STAGE OF THE RESEARCH

6.1 A First Prototype

As mentioned in the methodology section we begin our work by implementing a first prototype using conventional MDE Tools. The example of IAS19 calculation is a simple example representing a real use case in which two disciplines interact together creating a redundant use of data: Human Resource Management and Actuarial Science. “Fig. 1” and “Fig. 2” show two class diagrams representing two short extracts from the two metamodels. The diagrams were designed using Graphical Modeling EcoreTools (EcoreTools, 2015).

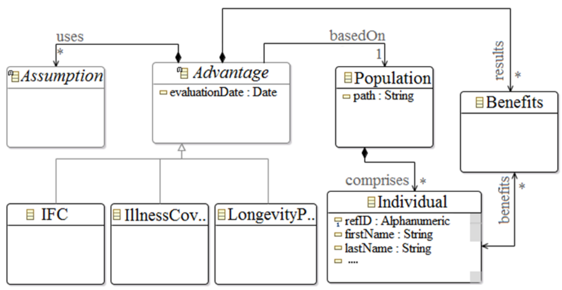

Figure 1: Class diagram representing an excerpt of the Actuarial study metamodel.

In collaboration with an Actuarial Cabinet (Actuaria Global, Casablanca, Morocco) we tried to implement a prototype of a DSMT dedicated to IAS 19 studies. We used classic modeling technologies to achieve it. The actuarial calculation starts from a data source (files, database...etc.) containing a company's employees information. This is provided by human resource service. The study ends with a report listing estimates of provisions to ensure the employee benefits at short, medium and long terms. Under IAS 19, the staff of a company benefits inter-alia of the three following benefits:

- Retirement Bonus (IFC).
- Longevity Pay.
- Illness Coverage.

The tool should provide the calculation of these three benefits.

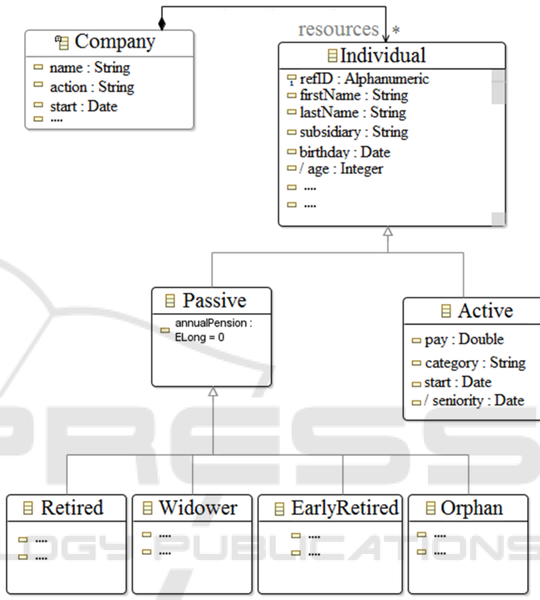

Figure 2: Class diagram representing an excerpt of the Human Resource metamodel.

6.1.1 Technical Architecture

Developed on the top of an Eclipse RCP base (The Eclipse project provides a complete set of tools for developing general purpose applications named Eclipse Rich Client Platform), the prototype is implemented with Java and public libraries under free license.

6.1.2 An MDE Approach

We applied an MDE approach to drive the development of the first prototype. We implemented a metamodel for each discipline (Human Resource and Actuarial) using Ecore tools (EcoreTools, 2015). Then we generated metamodels Java classes and adapters using the EMF Codegen facility (EMF, 2015). We implemented Java classes for the injection/extraction of data model employing POI

Apache Apis (Java API for Microsoft Documents) (Apache POI, 2015). We designed ergonomic interfaces using the Standard Widget Toolkit SWT (SWT, 2015) and Eclipse JFACE UI toolkit (JFace, 2015) in order to provide editors for different steps of the actuarial studies. The evaluation of benefits is done using a model transformation implemented using the Atlas Transformation Language ATL (JOUAULT, 2005). It produces a model (result) from which we can extract a textual report using a model to text generation implemented by Java classes.

6.1.3 Results & Discussion

This MDE approach gave us the means to implements a rich DSMT for the actuarial IAS 19 studies. A high-level abstraction helped us to better design the implemented prototype respecting a successful separation of interests. It is particularly important to separate the logical layer from the application layer, the presentation layer from the persistent layer and the data layer from that of their transformations. Through the application of MDE concepts, the prototype tool is composed of maintainable components, and the result is a modular application, more stable, more understandable, more adaptable, more reusable, more maintainable and easier to evolve.

Within this prototype tool, when the actuary begins his study for a given company, he chooses a specific advantage kind (IFC, Illness covertures, Longevity pay …etc.). Then the tool creates an instance of an empty model which is persisted in XMI. Afterward, the tool guides the actuary to complete this model using a succession of graphical interfaces. The first step consists of importing a population of individuals (active employees, retired employees, widowers, orphans…etc.) from a data source provided by the company (Excel files). Then the actuary proceeds to the verification and validation of data. Once he has finished, he creates the needed demographic and financial assumptions. After that, he configures its calculation methods and finally runs the evaluation of benefits. The result is the calculation of aggregate benefits consisting on global figures, but also individual figures for each individual in the population.

Unfortunately, despite all the advantages mentioned previously, this tool suffers from certain limits. We would have liked to be able to share a population of individuals between several actuarial studies. For each individual of this population, the actuarial study should associate, after calculation, some additional data (benefits). In some other cases, during the data validation, the actuary may have to modify a property's value of a given individual. Thus, he would like that this change impacts all studies implying this individual. Similar cases can be invoked for sharing assumptions, calculation methods ... etc.

Not finding a models composition mechanism providing necessary agility to implement these requirements, we choose to duplicate some properties of the *Individual* Human Resource metamodel class into the Actuarial metamodel *Individual* class. Thereby, each time the actuary imports the same employee data, for a new study, he creates a redundancy of data. Similarly, the operations of reusing a study configuration, assumptions or calculation methods, share the same inconvenient. In addition, to calculate other types of benefits or those of coming years he can not directly point some of his older configurations (present in the older studies). If we have found a models composition tool meeting the requirements of our use case, we would have designed the prototype architecture differently. We would have defined the assumptions into a separate metamodel, as well as for the calculation methods and so on. This would have allowed a better separation of concerns.

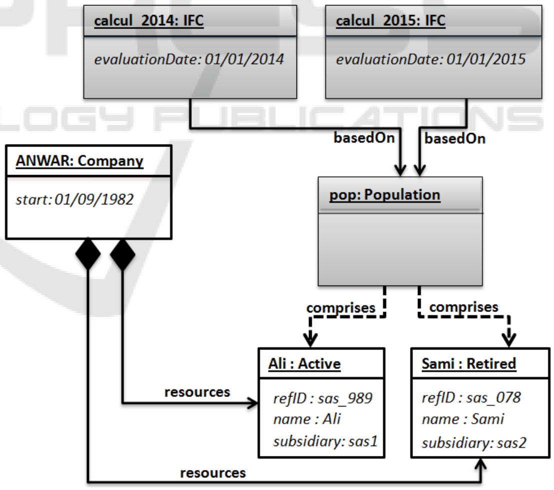

Figure 3: Representation of instances of Human Resource and Actuarial metamodels.

Let's focus on the sharing of population individuals for example. We can redefine our metamodels with a class Population of Actuarial metamodel that directly point the class Individual of the Human Resource one. The "Fig. 3" illustrates three instances of models conforming to these metamodels (two instances of Actuarial models and one insatnce of Human Resource model). We can see that the IFC study evaluated at 2014 is based on data of the ANWAR company employees without duplicating them on its own instance. The actuarial study of next year (2015) will reuse the same company's data. Furthermore, if the situation of an individual of this company, in the meantime, has been changed like if he has been transferred to another subsidiary, the change is automatically taken in account in the two IFC studies. Moreover, this can be generalized in the same way for assumptions, calculation methods…etc.

6.2 Solution's First Steps

We started the implementation of the first two modules: KM4M and CG4M. To create the KM4M language and its rich editor, we used the XText framework (XText, 2015); a framework for development of programming languages and domain-specific languages. Then we use the XPand framework (Xpand, 2015) to implement the CG4M module.

The KM4M language defines a simple grammar that translates the concepts of EMOF 2.5. In addition, it introduces new concepts to express relations between different metamodels; under three possible forms: equivalence relation, specialization relation, generalization relation. The first type is used to reuse a class from a metamodel "A" in a class of a metamodel "B" exactly as it is. The two other types of relations allow reusing a class defined in a metamodel "A" in a metamodel "B" by expanding or reducing its properties.

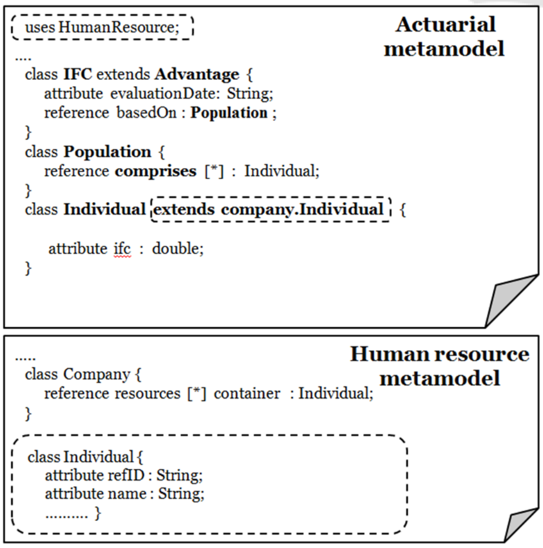

Figure 4: An example of two metamodels definition using the KM4M language and the specialisation relation concept.

The "Fig. 4" shows an example of a specialization relation applied to the use case presented above. In this example the class *Population* of the Actuarial metamodel reuses the class *Individual* of the Human Resource metamodel by expanding it with an additional attribute (representing the IFC benefit value).

7 CONCLUSION

This work tries to contribute to MDE tools with a new solution for multidisciplinary models composition. We presented in this paper a summarized view of our research work, the expected outcome and the stage of progress. Through a concrete use case, we have shown the interest of such a solution. We intend to continue the implementation of the three modules: KM4M, GM4M and MR4M. We then plan to prove our approach by considering the use case again and rebuild the first prototype using our proposition. Finally, we will compare the two prototypes by measuring the progress accomplished and assess the road which lies ahead. However, we are aware that our solution provides a model composition mechanism that overlay EMF implementation of EMOF 2.5. Therefore, the composed models made by our solution will still correlated with the use of our framework. While an extension of the concepts of EMOF 2.5 in order to provide model composition will allow a direct implementing of model composition at the EMF level. This solution could be shared among all the tools that are based on EMF at by the same time enabling interoperability between them.

REFERENCES

Mellor, S. J., Clark, A. N., Futagami, T., 2003. Model-Driven Development. In *Computer* (Vol. 20, no. 5, pp. 14-18). IEEE Computer Society.

France, R. Rump, B., 2007. Model-driven development of complex software: a research roadmap. In *Proceeding of FOSE'07: 2007 Future of Software Engineering* (pp. 37-54). IEEE Computer Society.

Shmidt, D.C., 2006. Guest editor's Introduction: model-driven engineering. In *Computer* (Vol. 39, no. 2, pp. 25–31). IEEE Computer Society.

Baccarini, D., 1996. The concept of project complexity - a review, In *International Journal of Project Management*, (pp. 201-204).

EMF, Eclipse Modeling Framework, 2015, url: https://eclipse.org/modeling/emf/

MOF 2.5 Core Specification, OMG Document formal/2015-06-05, 2015, url: http://www.omg.org/ spec/MOF/2.5/PDF

Pattis, R.E., 1980. EBNF: A Notation to Describe Syntax. url : http: //www.cs.cmu.edu/~pattis/misc/ebnf.pdf.

XMI 2.5, XML Metadata Interchange, Version 2.5.1, OMG document formal/2015-06-07, 2015, url: http://www.omg.org/spec/XMI/2.5.1/PDF

Steinberg, D., Budinsky, F., Paternostro, M., Merks, E., 2009. *EMF – Eclipse Modeling Framework 2.0*, Addison-Wesley Professional, 2nd edition.

Bagnato, A., Malo, P., Trujillo ,S., Mendialdua, X., De Carlos, X., Brosse, E., Sadovykh, A., 2014. Flexible and Scalable Modelling in the MONDO Project: Industrial Case Studies. In *Proceedings of the 2014 Extreme Modeling Workshop* (pp. 42–51). ACM.

Fouquet, F., Nain, G., Morin, B., Daubert, E., Barais O, Plouzeau, N., Jézéquel, J., 2012. An Eclipse Modelling Framework Alternative to Meet the Models@Runtime Requirements. In *Model Driven Engineering Languages and Systems*. Springer.

Kolovos, D., Rose, M., Matragkas, N., Paige, F., Guerra, E., Cuadrado, J., De Lara, J., Ráth, I., Varró, D., Tisi, M., Cabot, J., 2013. A research roadmap towards achieving scalability in model driven engineering. In *Proceedings of the Workshop on Scalability in Model Driven Engineering* (BigMDE '13). ACM.

Barmpis, K., Kolovos, D.S.,2012. Comparative analysis of data persistence technologies for large-scale models. In *Proceedings of the 2012 Extreme Modeling Workshop* (pp. 33–38). ACM.

Benelallam, A., G´omez, A., Suny´e, G., Tisi, M., Launay, D., 2014. Neo4EMF, a scalable persistence layer for EMF models. In *Modelling Foundations and Applications*, *of Lecture Notes in Computer Science* (Vol. 8569, pp. 230-241). Springer.

Garmendia, A., Guerra, E., Kolovos, D., De Lara, J., 2014. EMF Spliter: A Structured Approach to EMF Modularity. In *Proceedings of the 2014 Extreme Modeling Workshop* (pp. 22–31). ACM.

Kolovos, D., 2009. Establishing Correspondences between Models with the Epsilon Comparison Language. In *Proceedings of the 5th European Conference on Model Driven Architecture - Foundations and Applications* (pp. 146-157). Springer.

Kolovos, D., Paige, F., Polack, F., 2008. The epsilon transformation language. In *Theory and practice of model transformations: First International Conference, ICM* (pp. 46-60). Springer.

Fleurey, F,. Baudry, B., France, R., Ghosh, S., 2007. A Generic Approach for Automatic Model Composition. In *H. Giese, MoDELS Workshops, of Lecture Notes in Computer Science* (Vol. 5002, pp. 7-15). Springer.

Didonet Del Fabro, M., Valduriez, P., 2009. Towards the Efficient Development of Model Transformations using Model Weaving and Matching Transformations. In *Software and Systems Modeling* (Vol. 8, No. 3, pp. 305–324). Springer.

Clasen, C., Jouault, F., Cabot, J., 2011. Virtual Composition of EMF Models. In *7èmes Journées sur l'Ingénierie Dirigée par les Modèles (IDM 2011)*. unpublished.

IASB, IAS 19 amendments, 2004. Employee benefits : actuarial gains and losses, group plans and disclosures.

EcoreTools, 2015, url: http://www.eclipse.org/ecoretools/

Apache POI, The Java API for Microsoft Documents, 2015, url: https://poi.apache.org/

SWT,The Eclipse Standart Widget Toolkit, 2015, url: https://www.eclipse.org/swt/

JFACE, The JFace UI framework, 2015, http://help.eclipse.org/juno/index.jsp?topic=/org.eclipse.platform.doc.isv/guide/jface.htm

JOUAULT, F., KURTEV, I., 2005. Transforming Models with ATL. In J.-M. BRUEL (Ed.), *MoDELS'05 Proceedings of the 2005 international conference on Satellite Events at the MoDELS* (Lecture Notes in Computer Science ed., Vol. 3844, pp. 128-138). Springer-Verlag Berlin, Heidelberg.

XText, 2015, url : http://www.eclipse.org/Xtext/

Xpand, Eclipse Model to Text (M2T) Xpand , 2015, url : http://www.eclipse.org/modeling/m2t/